Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Previously Presented) A process for translating instructions belonging to a first set of instructions that are pipelined scalar processor instructions into instructions belonging to a second set of instructions that are VLIW processor instructions for execution on a VLIW processor that includes a core, said process comprising the following operations:

providing a first set of registers corresponding to the instructions of said first set of instructions;

providing a second set of registers corresponding to the instructions of said second set of instructions;

mapping each register of said first set of registers in a corresponding register of said second set of registers designed to emulate the behavior of the register of said first set of registers, performing a unique independent translation of the instructions of said first set of instructions into said second set of instructions;

said operations of providing the second set of registers and of mapping being obtained by adding functional units to VLIW processor and keeping said core unaltered; and

performing said translation in the absence of direct access to resources of said core.

2. (Previously Presented) The process according to claim 1, wherein adding functional units includes adding a translation device external to said core of the VLIW processor,

said translation device intercepting accesses to a storage area reserved to the first set of instructions.

3. (Previously Presented) The process according to claim 2, further comprising forcing a program counter of the VLIW processor to point to a translation memory reserved in the translation device for containing a translation of an instruction belonging to the first set of instructions, and decoding the instruction into a decoded instruction.

4. (Previously Presented) The process according to claim 3, further comprising loading in said translation memory all instructions that constitute the translation of the decoded instruction.

5. (Previously Presented) The process according to claim 4, further comprising associating said all instructions that constitute the translation of the decoded instruction to a jump-to-link to a next instruction of the first set of instructions to be executed, so that all instructions of the first set of instructions that are not directly mappable to instructions of the second set of instructions entail a jump to the translation memory and the jump-to-link for loading the next instruction of the first set of instructions.

6. (Previously Presented) The process according to claim 1, further comprising translating all instructions of the first set of instructions to which there does not correspond an equivalent single instruction in the second set of instructions into an unconditioned GOTO jump.

7. (Previously Presented) The process according to claim 1, further comprising an operation of forcing a program counter of the VLIW processor to operationally emulate a program counter of a pipelined scalar processor.

8. (Previously Presented) The process according to claim 7, wherein said operation of forcing the program counter of the VLIW processor includes forcing said program counter of the VLIW processor to contain a value that is the value of the program counter of the pipelined scalar processor upon loading an instruction belonging to the first set of instructions.

9. (Previously Presented) The process according to claim 8, further comprising executing one translated instruction and ending the executing by performing a jump to an address of a next instruction belonging to the first set of instructions.

10. (Previously Presented) The process according to claim 9, wherein emulating the program counter of the pipelined scalar processor includes using a counter register, incrementing said counter register so that each instruction that accesses said counter register during the execution step will have a behavior that is coherent with the program counter on the pipelined scalar processor, and decrementing said counter register for pointing to the next instruction belonging to the first set of instructions.

11. (Original) The process according to claim 10, wherein said counter register is incremented by a value eight and decremented subsequently by a value four.

12. (Previously Presented) The process according to claim 11, wherein, for instructions belonging to the first set of instructions that have the program counter of the VLIW processor as a destination, a pointing to the next instruction belonging to the first set of instructions is obtained by loading into a link register of the VLIW processor an updated value of the counter register and by making an unconditioned GOTO link jump.

13. (Previously Presented) The process according claim 1 wherein adding functional units includes adding a translation device external to said core of the VLIW processor, said translation device intercepting accesses to a storage area reserved to the first set of instructions, the process further comprising allowing a program counter of the VLIW processor to evolve freely in the absence of jumps.

14. (Previously Presented) The process according to claim 13, wherein said translation device is designed to execute operations of intercepting accesses to the storage area reserved to the first set of instructions and controlling a set of pointer registers for deciding

whether to execute subsequent instructions as instructions of the second set of instructions or as instructions of the first set of instructions.

15. (Previously Presented) The process according to claim 14, wherein said translation device is inactive until the core of the VLIW processor executes instructions belonging to said second set of instructions and refers an accesses-to-memory to an instruction cache memory, and said translation device is activated when there is an access to the storage area reserved to the first set of instructions.

16. (Previously Presented) The process according to claim 15, wherein when said translation device is activated, the translation device loads into a selected one of its internal registers belonging to the set of pointer registers an address which is accessed and carries out reading of a corresponding instruction from the storage area.

17. (Previously Presented) The process according to claim 16, further comprising the operations of: translating said corresponding instruction read from the storage area and storing said corresponding instruction;

allocating an execution window, to which are referred all accesses to memory addresses that start from a current value of the program counter of the VLIW processor and cover an area equal to one occupied by the translation;

reading, using the core of the VLIW processor, a first instruction of the translation from the storage area reserved to the first set of instructions in the translation device;

incrementing a selected register by a value four, to point to the next instruction of the first set of instructions; and

closing said execution window after reading a last instruction of the translation, and, if at a next access to memory the selected register points outside the storage area reserved to the first set of instructions, deactivating the translation device.

18. (Previously Presented) The process according to claim 17, further comprising, in the presence of jumps in a program consisting of instructions of the first set of instructions, rewriting said selected register by a store-word operation contained in the translation of an instruction of the first set of instructions into instructions of the second set of instructions.

19. (Original) The process according to claim 1 wherein the operation of translation on the VLIW processor starts with verification of an execution condition, which includes evaluating one or more flags present in a status register.

20. (Currently Amended) A translator device for translating instructions belonging to a first instruction set that are pipelined scalar processor instructions into instructions belonging to a second instruction set that are VLIW processor instructions for execution on a VLIW processor that includes a core, said translation device comprising

a translation subsystem designed to receive at input a instruction of the first instruction set and supply at output a translation including one or more instructions of the second instruction set;

a translation memory coupled to the translation subsystem and structured to store said translation; and

a control device for taking said translation from said translation memory and supplying it to the core of said VLIW processor, wherein the device is connected between the core of said VLIW processor and an instruction-cache memory of said VLIW processor, which operates on a first memory containing instructions of the first instruction set and a second memory containing instructions of the second instruction set, and wherein the control device intercepts accesses of the core of the VLIW processor to said first and second memories.

21. (Original) The device according to claim 20, wherein said translation subsystem operates based on a code table stored in said translation memory.

22. (Canceled)

23. (Currently Amended) The device according to claim ~~22~~20, further comprising a set of pointer registers at least in part designed for controlling access to said memories.

24. (Previously Presented) A computer-readable medium having contents that, when loaded into a computer, cause the computer to perform a process for translating instructions belonging to a first set of instructions that are pipelined scalar processor instructions into instructions belonging to a second set of instructions that are VLIW processor instructions for execution on a VLIW processor that includes a core, said process comprising the following operations:

providing a first set of registers corresponding to the instructions of said first set of instructions;

providing a second set of registers corresponding to the instructions of said second set of instructions;

mapping each register of said first set of registers in a corresponding register of said second set of registers designed to emulate the behavior of the register of said first set of registers, performing a unique independent translation of the non-native instructions of said first set of instructions into said second set of instructions;

said operations of providing a second set of registers and of mapping being obtained by adding functional units to the VLIW processor and keeping said core unaltered; and

performing said translation in the absence of direct access to resources of said core.

25. (Previously Presented) The process according to claim 24, wherein emulating a program counter of the pipelined scalar processor includes using a counter register, incrementing said counter register so that each instruction that accesses said counter register during the execution step will have a behavior that is coherent with a program counter on a

pipelined scalar processor, and decrementing said counter register for pointing to the next instruction belonging to the first set of instructions.

26. (Previously Presented) The process according to claim 25, wherein said counter register is incremented by a value eight and decremented subsequently by a value four.

27. (Previously Presented) The process according to claim 16, further comprising the operations of: translating said corresponding instruction read from the storage area and storing said corresponding instruction;

allocating an execution window, to which are referred all accesses to memory addresses that start from a current value of the program counter of the VLIW processor and cover an area equal to one occupied by the translation;

reading, using the core of the VLIW processor, a first instruction of the translation from the storage area reserved to the first set of instructions in the translation device;

incrementing a selected register by a number equal to a length in bytes of each instruction of the first set of instructions; and

closing said execution window after reading a last instruction of the translation, and, if at a next access to memory the selected register points outside the storage area reserved to the first set of instructions, deactivate the translation device.

28. (Previously Presented) A process for translating instructions belonging to a first set of instructions that are pipelined scalar processor instructions into instructions belonging to a second set of instructions that are VLIW processor instructions for execution on a VLIW processor that includes a core, said process comprising the following operations:

providing a first set of registers corresponding to the instructions of said first set of instructions;

providing a second set of registers corresponding to the instructions of said second set of instructions;

mapping each register of said first set of registers in a corresponding register of said second set of registers designed to emulate the behavior of the register of said first set of registers, performing a unique independent translation of the instructions of said first set of instructions into said second set of instructions;

said operations of providing the second set of registers and of mapping being obtained by adding functional units to VLIW processor and keeping said core unaltered; and

performing said translation in the absence of direct access to resources of said core;

wherein adding functional units includes adding a translation device external to said core of the VLIW processor, said translation device intercepting accesses to a storage area reserved to the first set of instructions, the process further comprising allowing a program counter of the VLIW processor to evolve freely in the absence of jumps;

translating said corresponding instruction read from the storage area and storing said corresponding instruction;

allocating an execution window, to which are referred all accesses to memory addresses that start from a current value of the program counter of the VLIW processor and cover an area equal to one occupied by the translation;

reading, using the core of the VLIW processor, a first instruction of the translation from the storage area reserved to the first set of instructions in the translation device;

incrementing a selected register by a number equal to a length in bytes of each instruction of the first set of instructions; and

closing said execution window after reading a last instruction of the translation, and, if at a next access to memory the selected register points outside the storage area reserved to the first set of instructions, deactivate the translation device.

29. (Previously Presented) The process according to claim 28, further comprising, in the presence of jumps in a program consisting of instructions of the first set of instructions, rewriting said selected register by a store-word operation contained in the

translation of an instruction of the first set of instructions into instructions of the second set of instructions.

30. (Previously Presented) The process according to claim 28, wherein said translation device is designed to execute operations of intercepting accesses to the storage area reserved to the first set of instructions and controlling a set of pointer registers for deciding whether to execute subsequent instructions as instructions of the second set of instructions or as instructions of the first set of instructions.

31. (Previously Presented) The process according to claim 30, wherein said translation device is inactive until the core of the VLIW processor executes instructions belonging to said second set of instructions and refers an accesses-to-memory to an instruction cache memory, and said translation device is activated when there is an access to the storage area reserved to the first set of instructions.